Network Monitoring as a Service (NMaaS)

NANOG 79, June 1st-3rd 2020

Bryan TO VAN TRANG – bryan.tovantrang@orange.com
Raquel RUGANI LAGE – raquel.ruganilage@orange.com
Anthony LAMBERT – anthony.lambert@orange.com

orange™

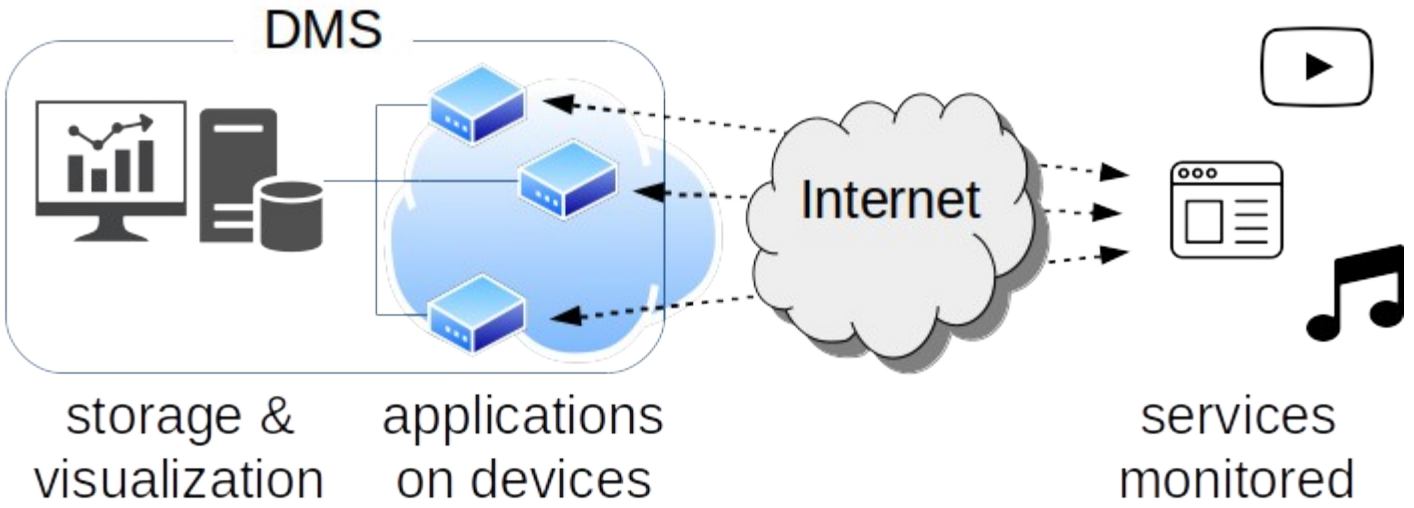
Agenda

- Context
- Overview
- Under the hood
- How to deploy
- Conclusion

Context

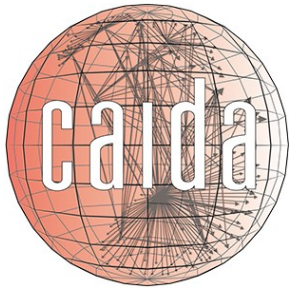
Network and services monitoring is crucial for quality and security.

This can explain the rise of Distributed Measurement Systems (DMS): devices deployed over networks, embedding monitoring applications periodically testing network and services and retrieving measurements further used for dashboarding, alerting, etc.



Context

Examples of DMS range from private infrastructures deployed by ISPs to measure end users « QoE » (e.g. SamKnows, IpLabel, home made) to large scale public infrastructures (e.g. RIPE Atlas, CAIDA Ark) that can be used for Internet Tomography studies.

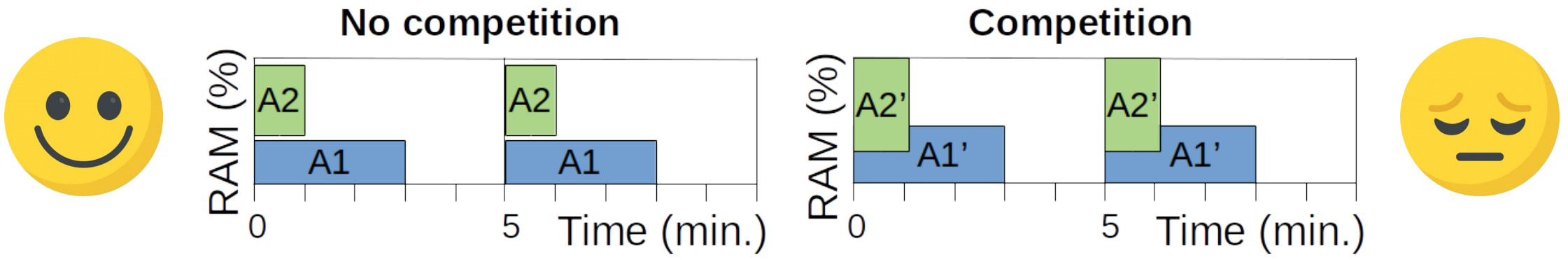


And more …

Context

Designing DMS is challenging especially as they must scale and provide reliable measurements.

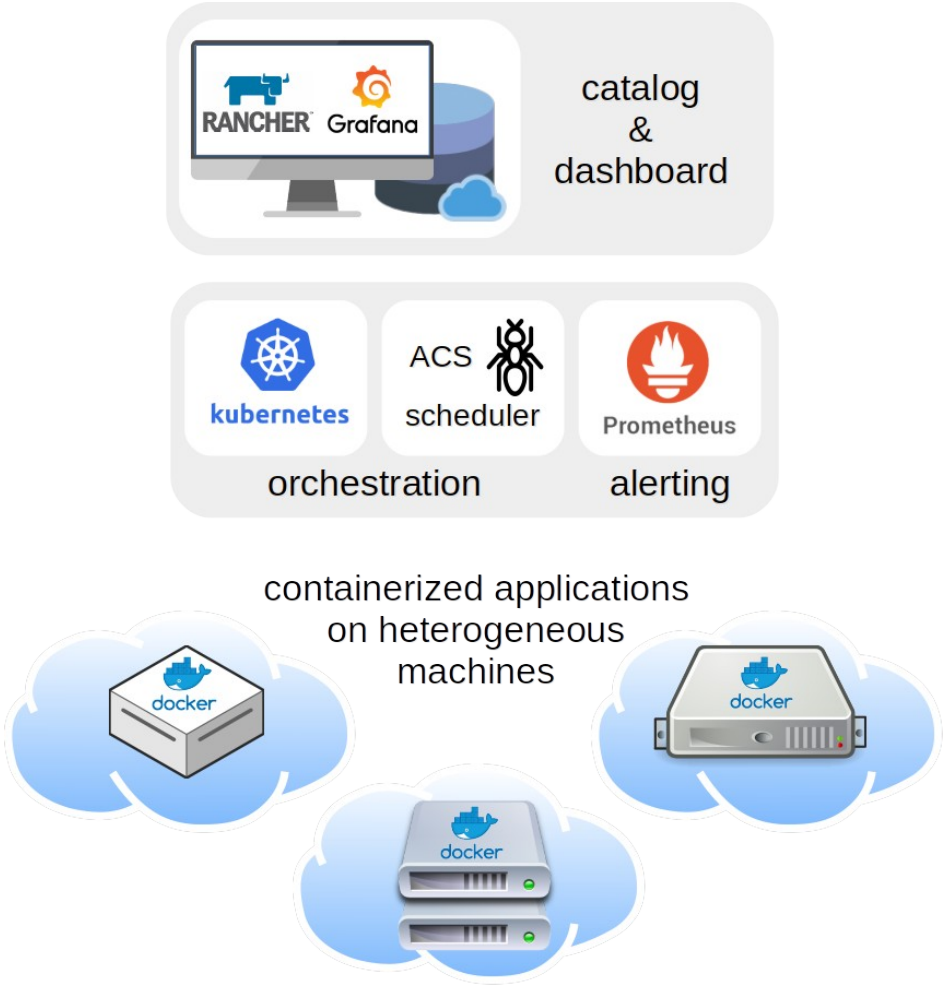
Especially, when many applications are collocated on the same machines, one has to make sure they do not compete for resources while executing as to not bias the collected measurements.



Context

To this end, we propose the NMaaS, an open-source platform, publicly available which enables to deploy and manage containerized measurement applications on a pool of physical machines.

Furthermore, we add a scheduler to the NMaaS to ensure applications do not compete for resources.

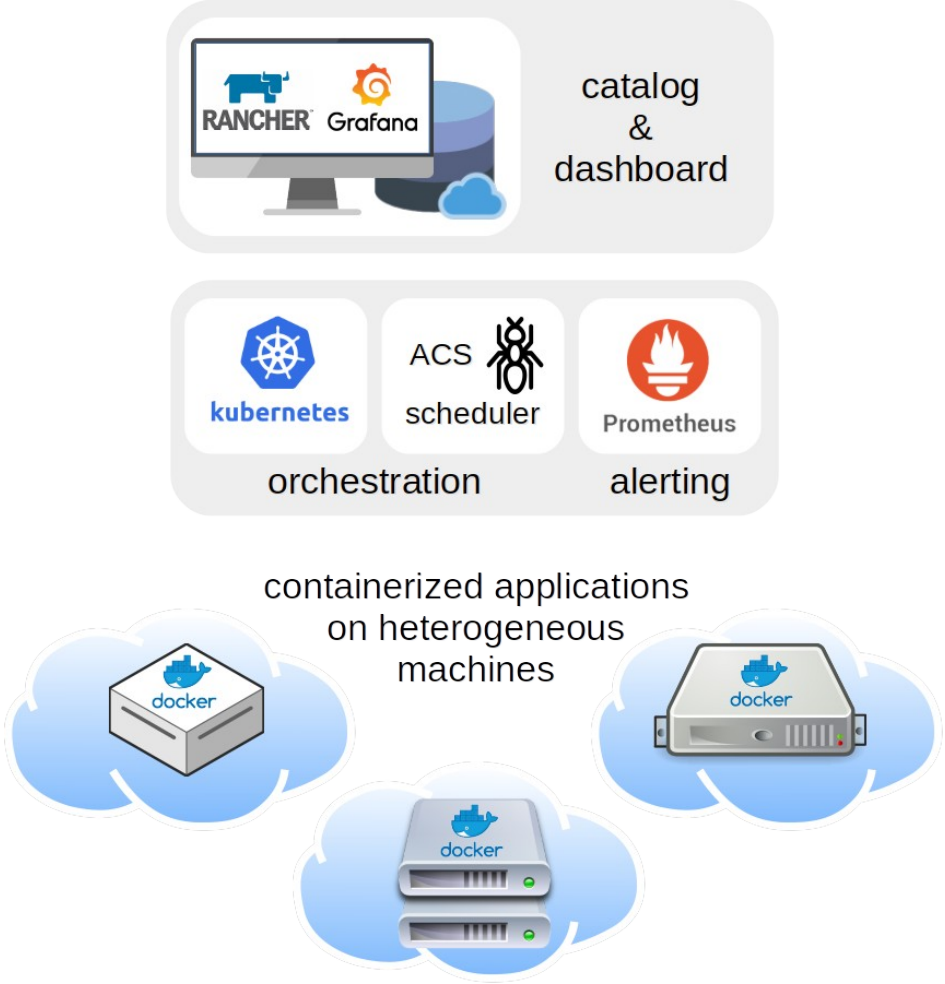


https://github.com/Orange-OpenSource/NMaaS

Context

Two main use-cases:

- Private DMS, people can download an NMaaS instance and deploy it on nodes in their network(s) to measure performances of their network(s) and services;
- Public DMS, people can download an NMaaS instance and deploy it on nodes over potentially multiple networks to set up a large scale public DMS.

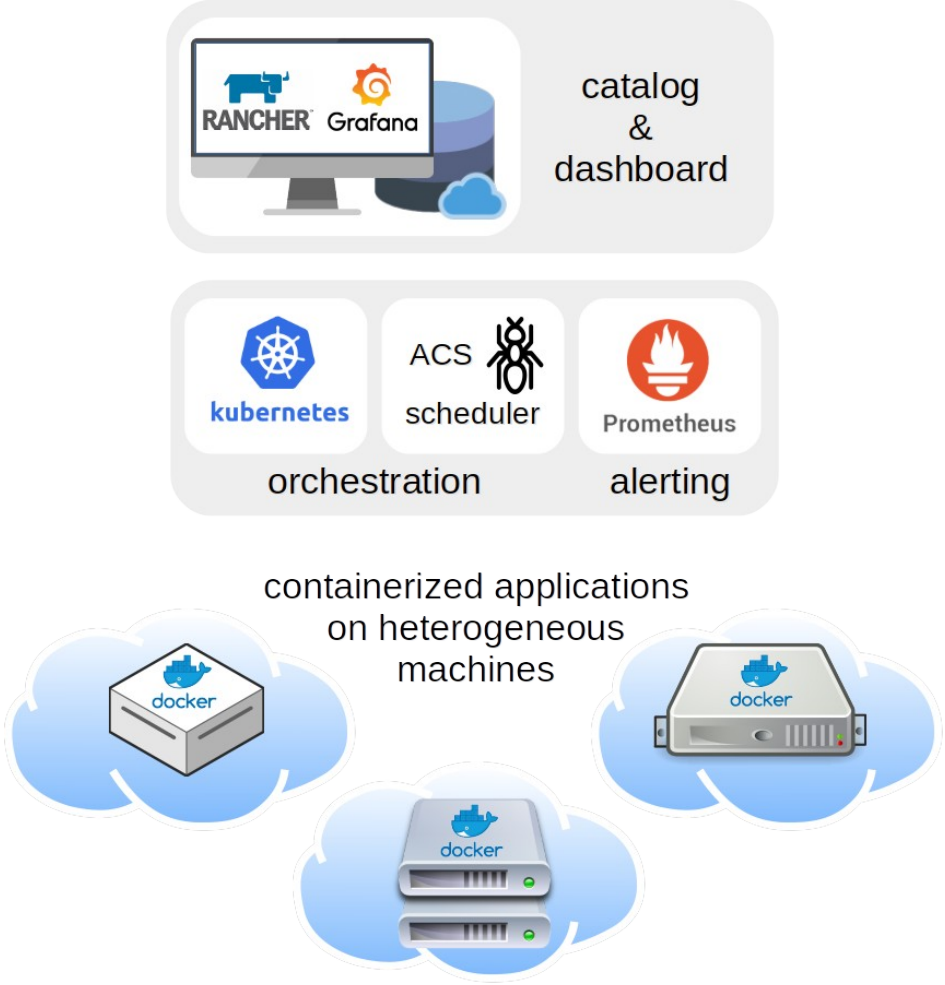


https://github.com/Orange-OpenSource/NMaaS

Context

An NMaaS instance is accessible via an online application allowing to:

- choose monitoring applications from a pre-defined catalog to be deployed on machines in the network;
- visualize and manage their pool of machines, as well as the monitoring applications deployed on them;
- examine the results of the measurements and alerts raised.



https://github.com/Orange-OpenSource/NMaaS

Context

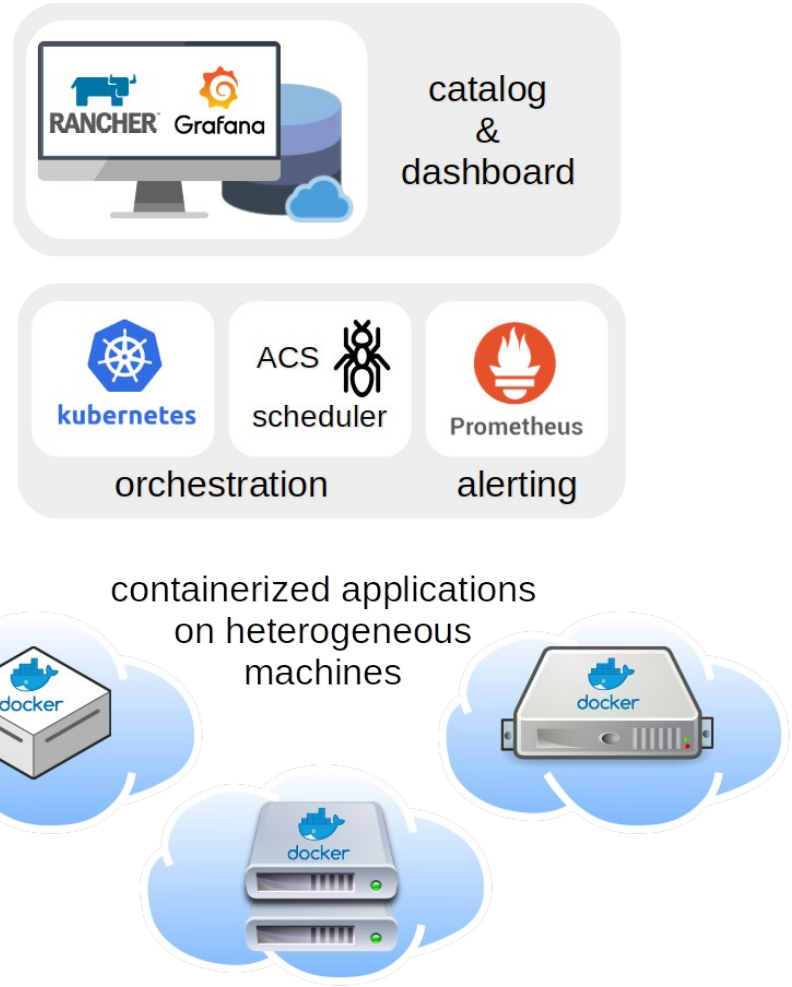
Main roles in a NMaaS environment:

User: registers for credentials to deploy apps and run tests.

Node Owner: deploys & manages physical devices in networks.

App Developer: submits apps to be added to the catalog.

Validator: checks for security on nodes, apps and users.



https://github.com/Orange-OpenSource/NMaaS

Agenda

Context

Overview

Under the hood

How to deploy

Conclusion

Overview: platform

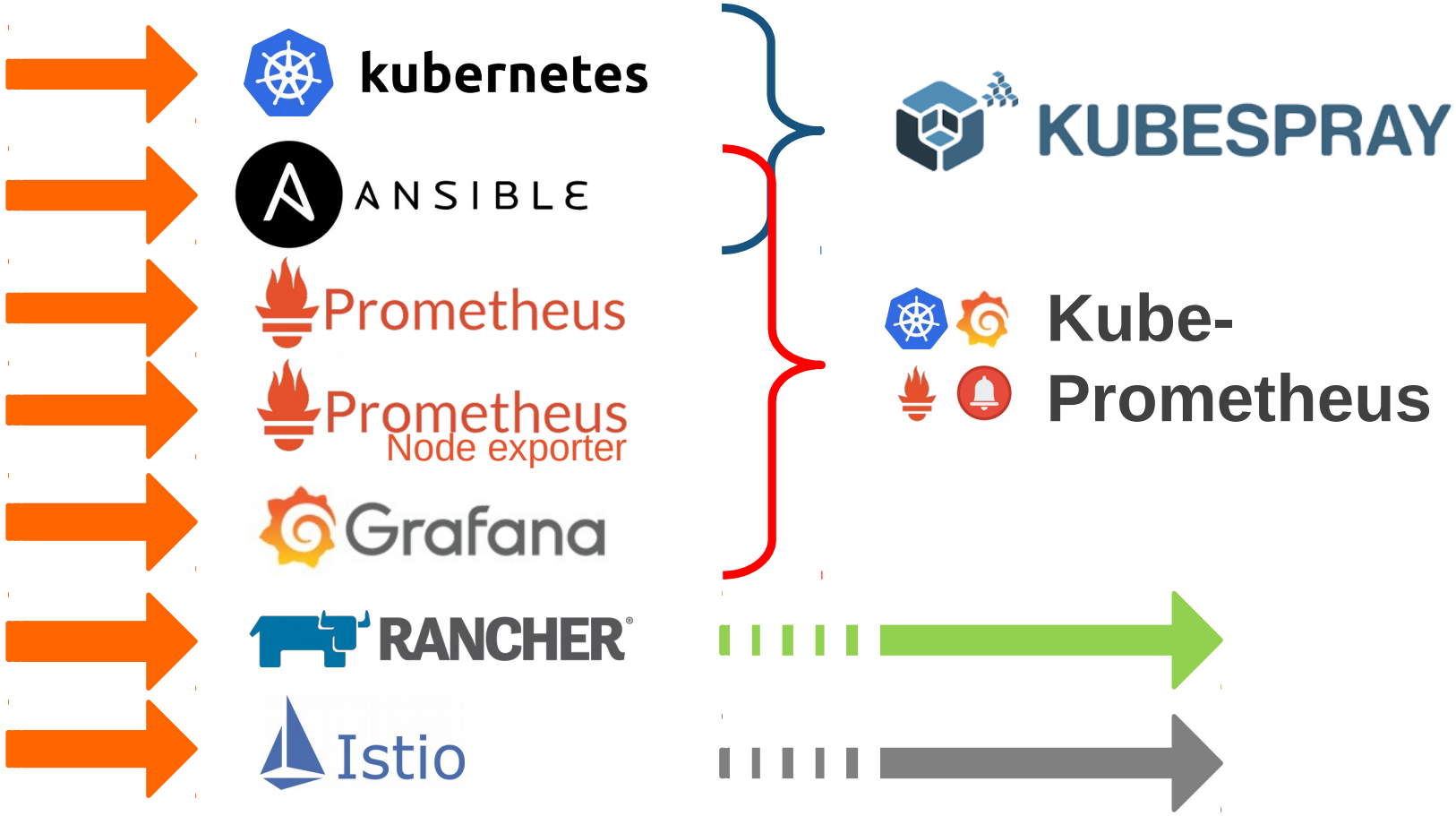
- Orchestration
- Automation
- Database
- Metric collection
- GUI
- Dashboard
- Middleware

Overview: platform

- Orchestration
- Automation
- Database
- Metric collection
- GUI
- Dashboard
- Middleware

Overview: platform

- Orchestration
- Automation
- Database
- Metric collection
- GUI
- Dashboard
- Middleware



Overview: platform

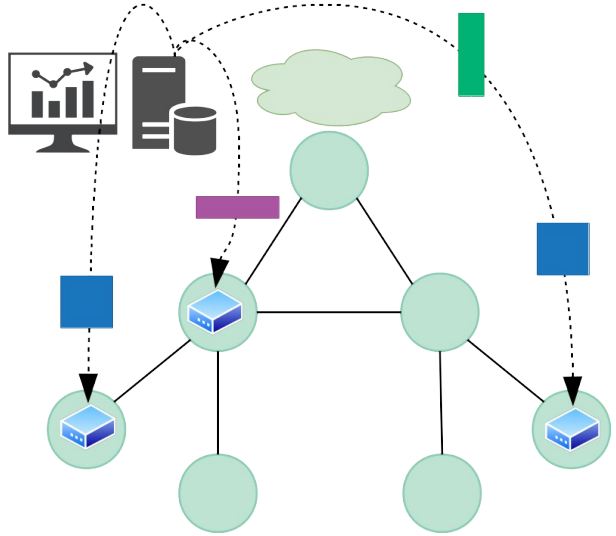
Additionnal features:

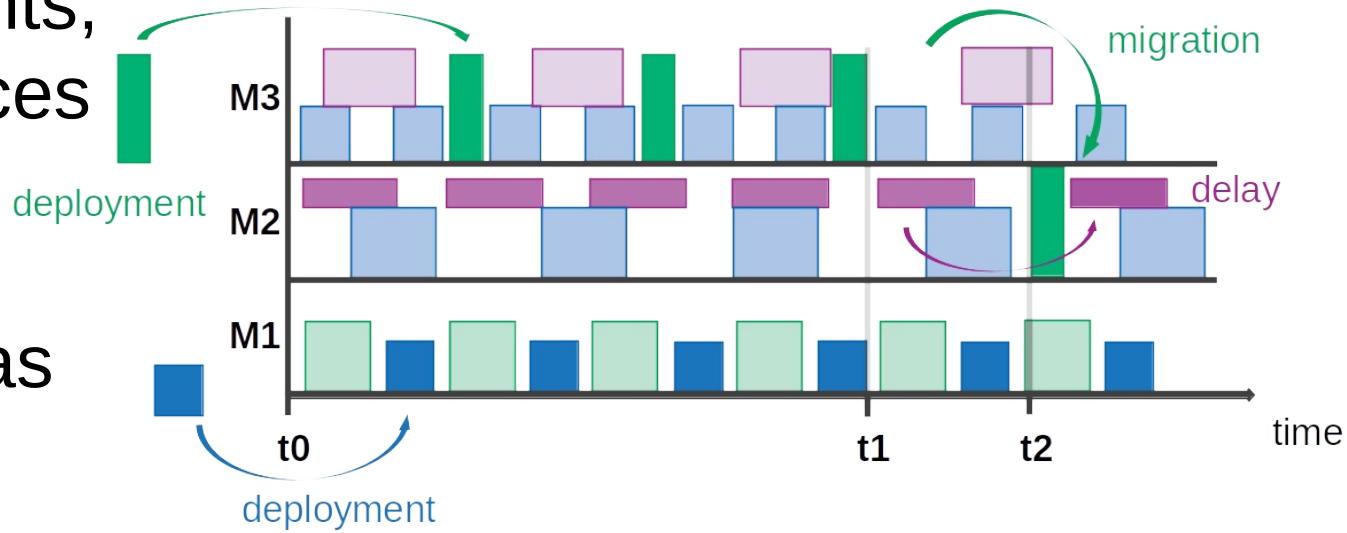
- KUBESPRAY → Private local repository
- Kube-Prometheus → AlertManager
- Istio → kiali

Overview: scheduler

We add a scheduler to the NMaaS to ensure applications do not compete for resources:

- based on an Ant Colony System (ACS) metaheuristic[1];
- handles requests for deployments;
- gets the current state of resources on devices;
- computes new resource allocations and test schedules as well as potential migrations.

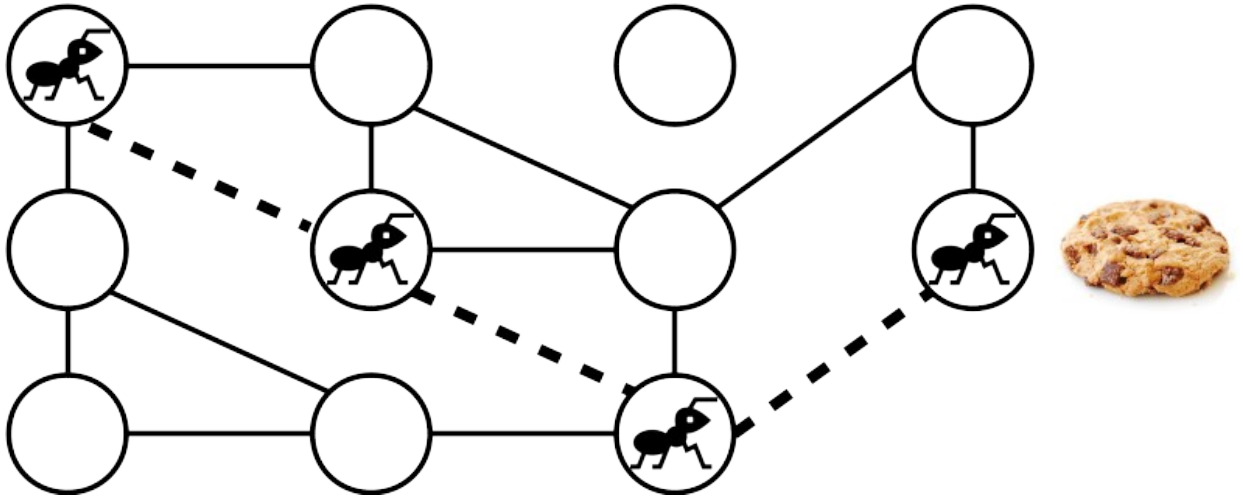




[1] Rugani Lage Raquel et al. « Allocating and Scheduling Measurement Applications » NOMS2020

Overview: scheduler

Natural behavior of food discovery of ants: ants deposit pheromone on their path to food, other ants then follow paths where pheromone concentration is higer leading the colony to converge to the shortest path.



ACS takes inspiration from this behavior to explore the research space and identify good solutions to optimization problems. An ant builds a solution by traversing a construction graph.

Overview: apps catalog

Easy and rapid integration of applications to the platform thanks to the catalog system and its container-based architecture.

First catalog contains:

- an IP spoofing detection app;
- a web (resp. streaming) QoS measure app;
- a web (resp. streaming) cartography app.

The goal is then to motivate users (You) to propose new apps to be added to the catalog.

Agenda

Context

Overview

Under the hood

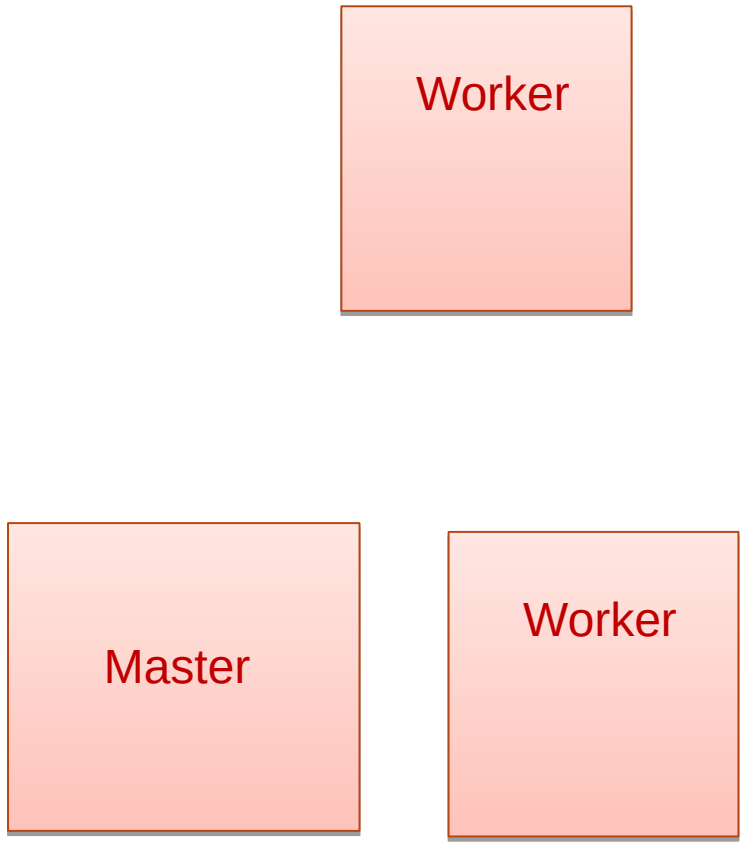
How to deploy

Conclusion

Under the hood (Macro overview)

Example:

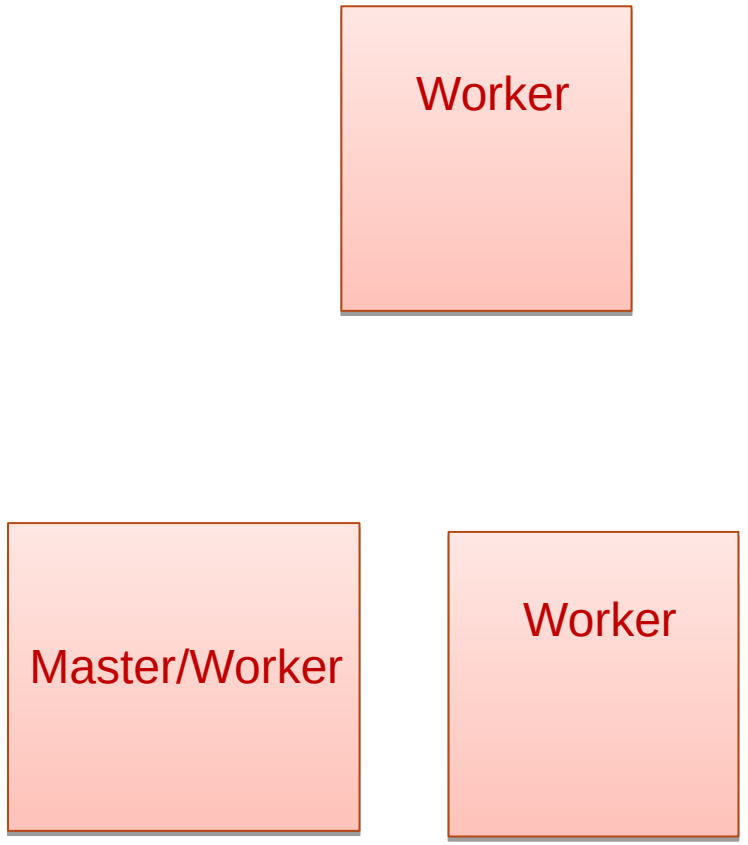
- 3 servers: 1 master and 2 workers



Under the hood (Macro overview)

Example:

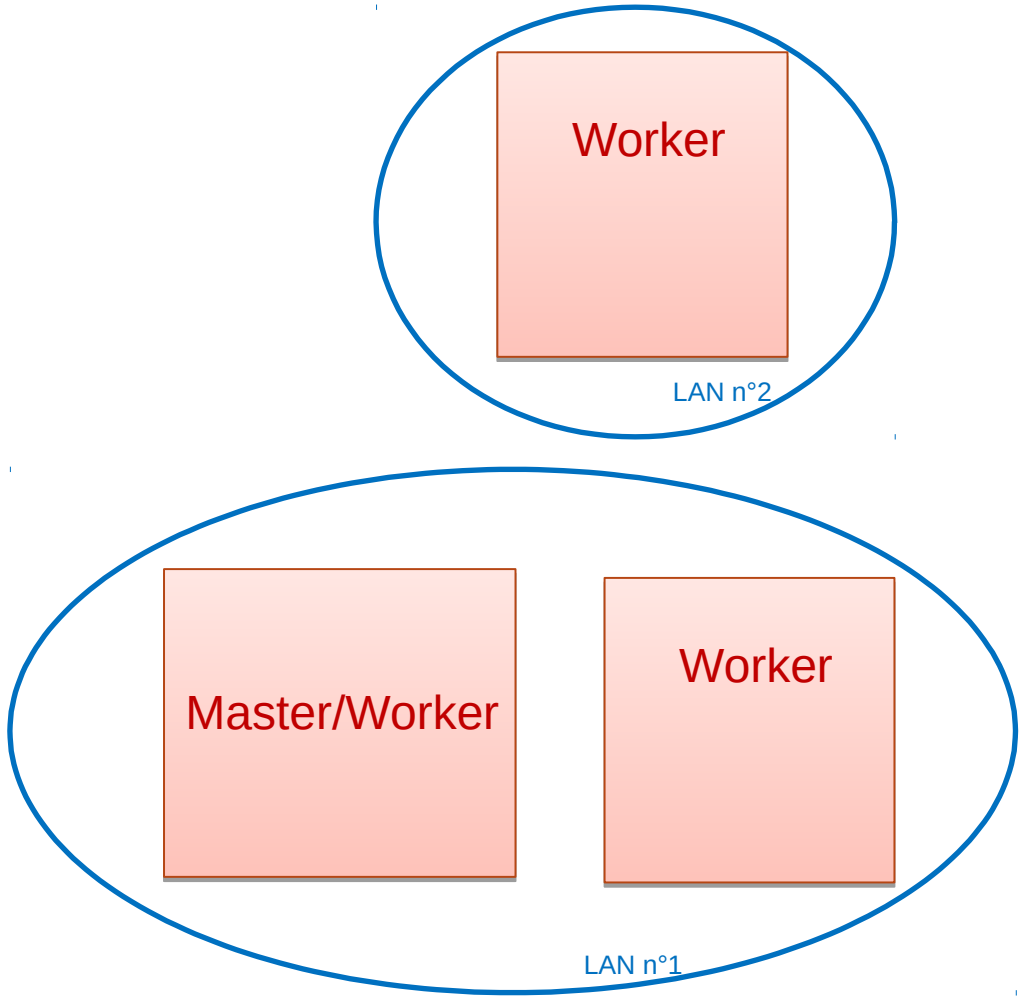
- 3 servers: 1 master and 2 workers
- The master can also be a worker



Under the hood (Macro overview)

Example:

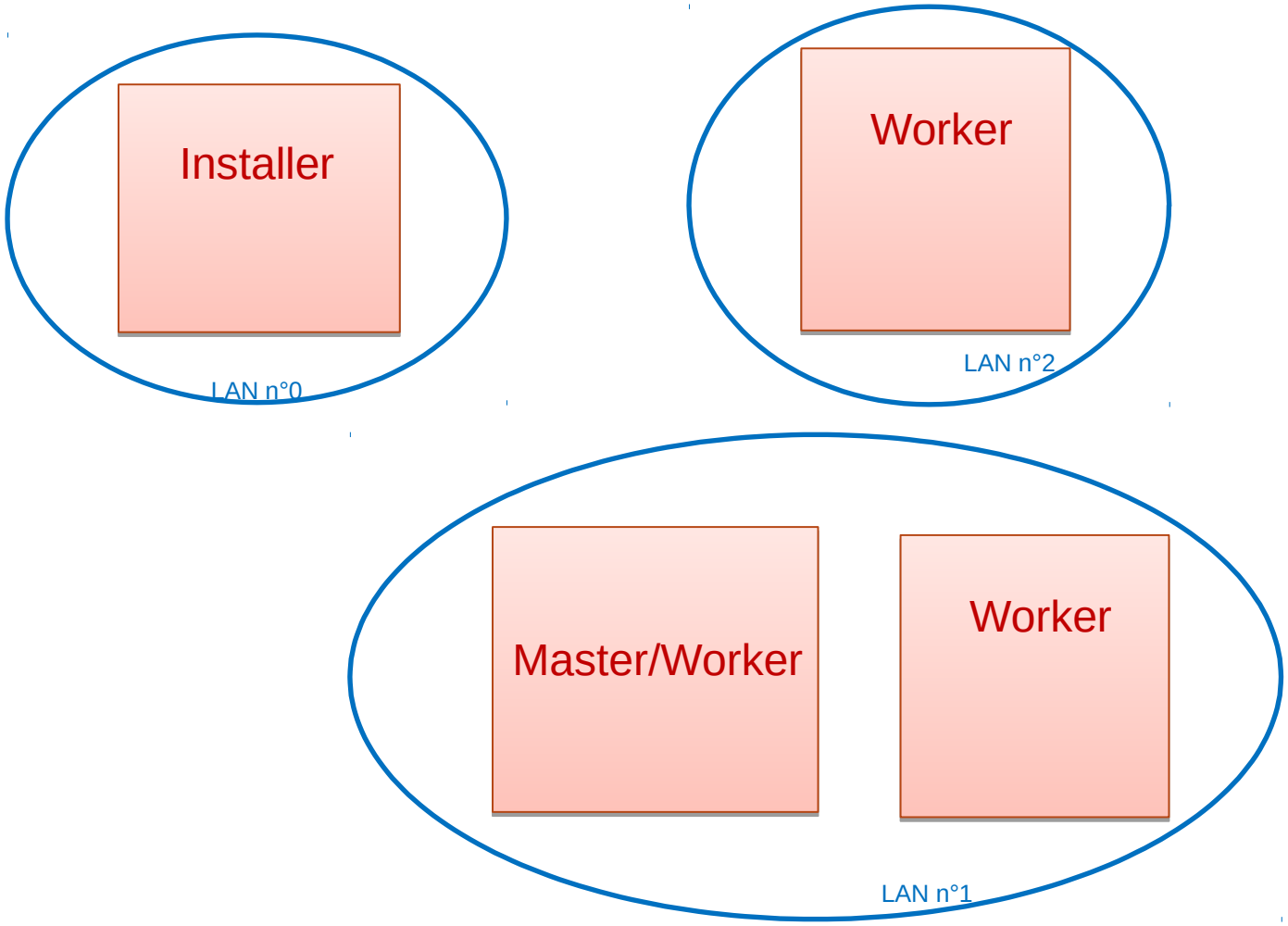
- 3 servers: 1 master and 2 workers
- The master can also be a worker
- Servers on same or different LAN



Under the hood (Macro overview)

Example:

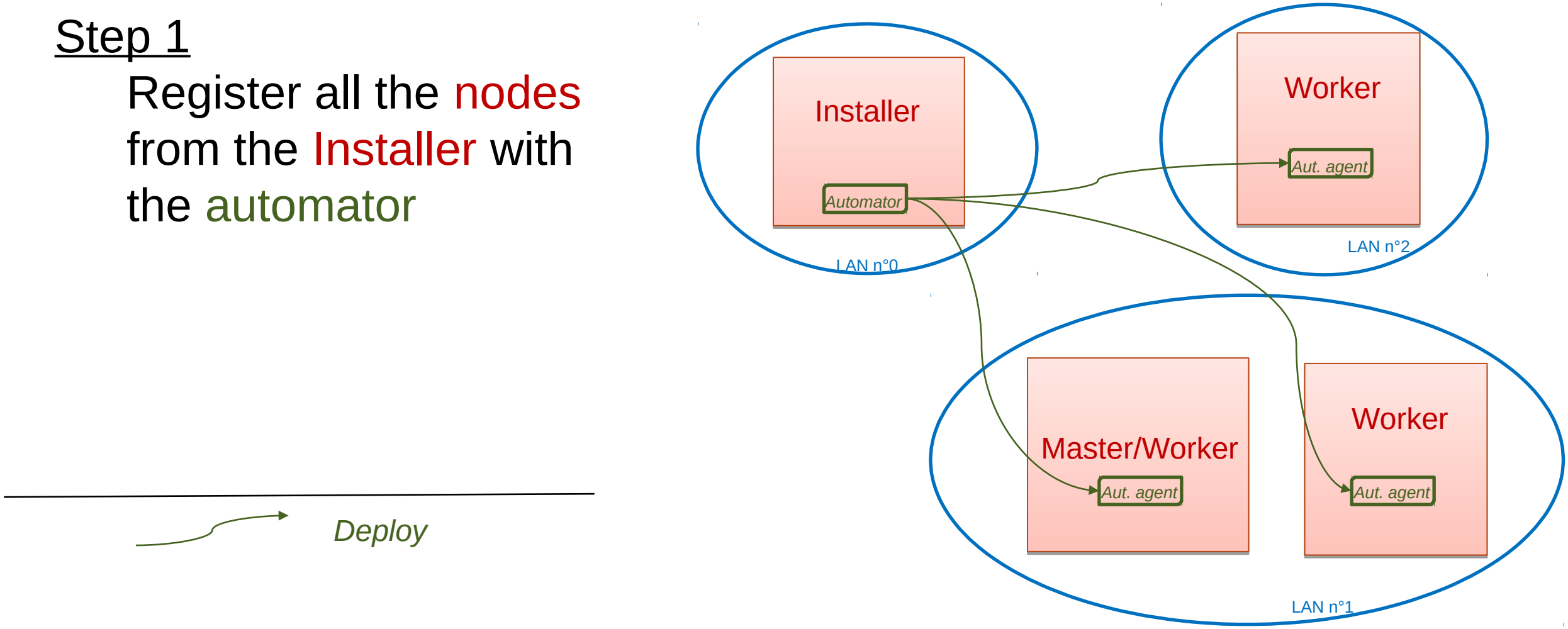
- 3 servers: 1 master and 2 workers
- The master can also be a worker
- Servers on same or different LAN
- Installing from your own machine



Under the hood (Macro overview)

Step 1

Register all the nodes from the Installer with the automator



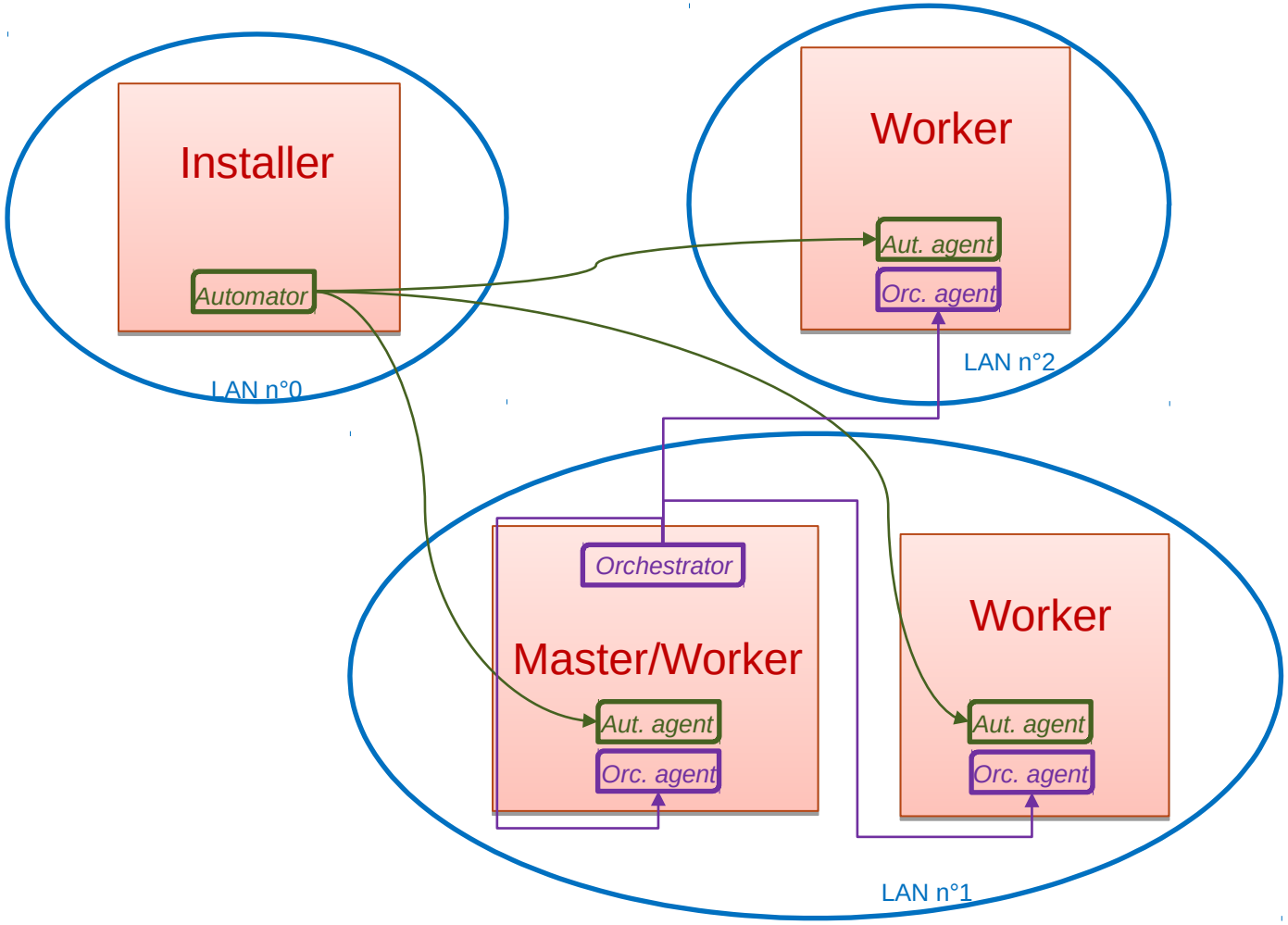
Under the hood (Macro overview)

Step 1

Register all the nodes from the Installer with the automator

Step 2

Initiate the cluster of the registered nodes with the orchestrator



Deploy

Control

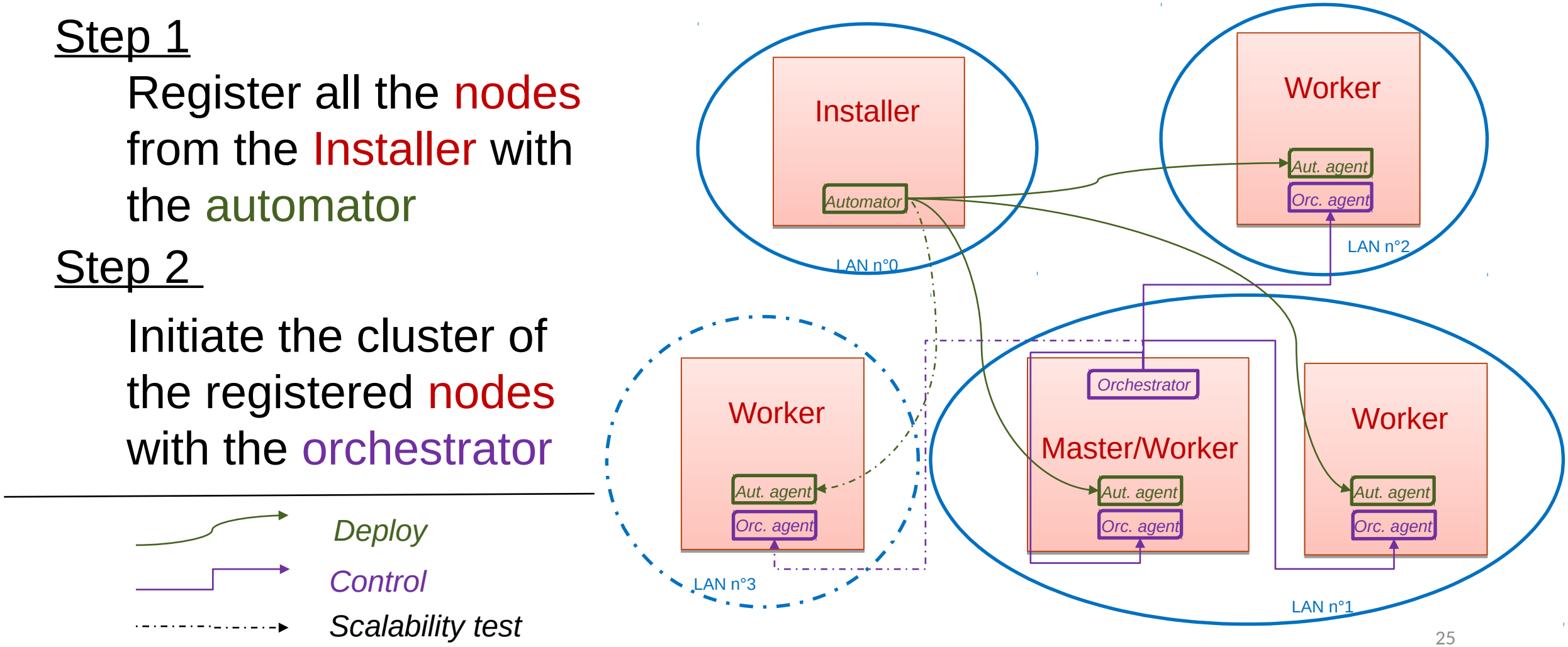
Under the hood (Macro overview)

Step 1

Register all the nodes from the Installer with the automator

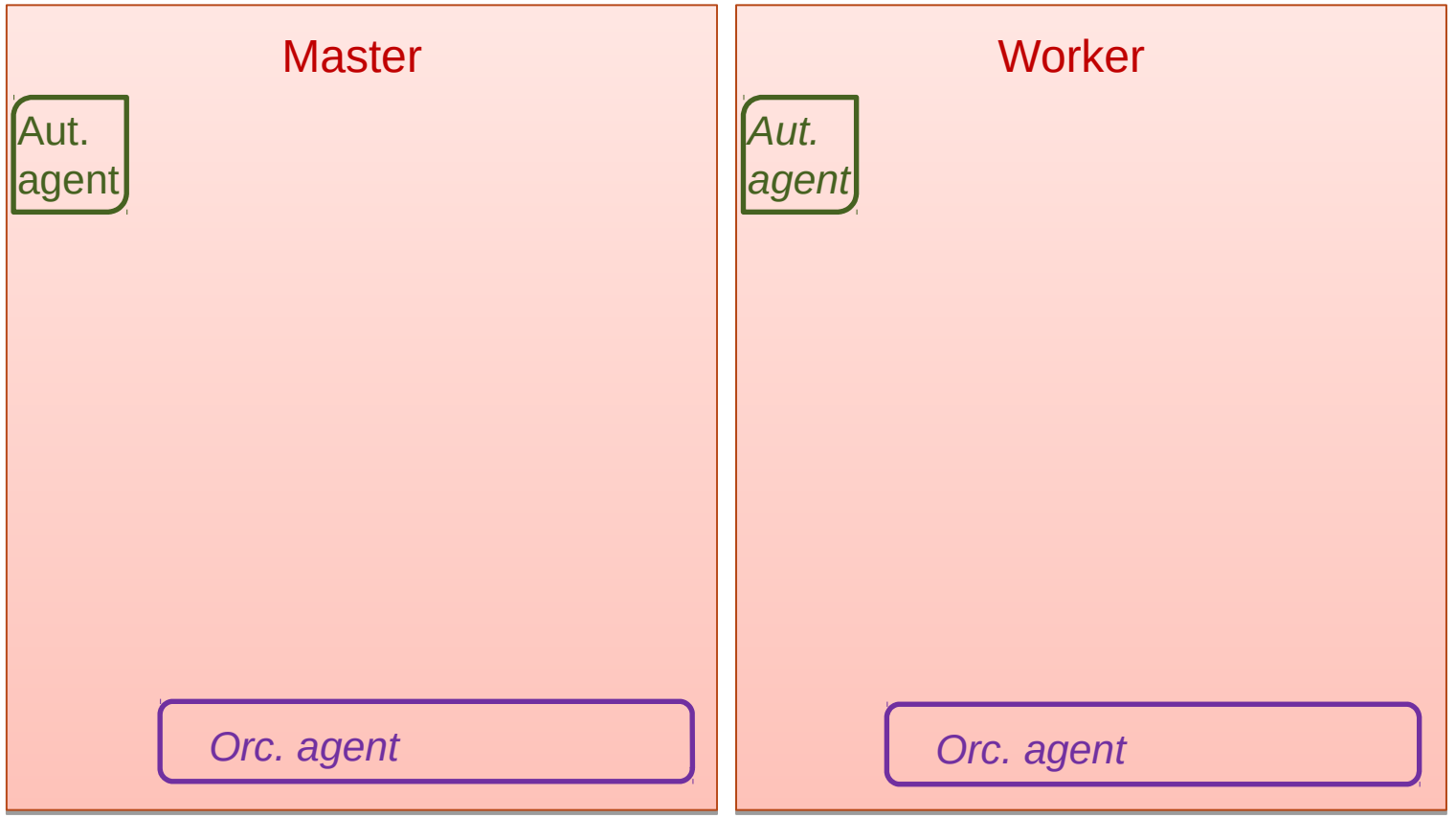
Step 2

Initiate the cluster of the registered nodes with the orchestrator



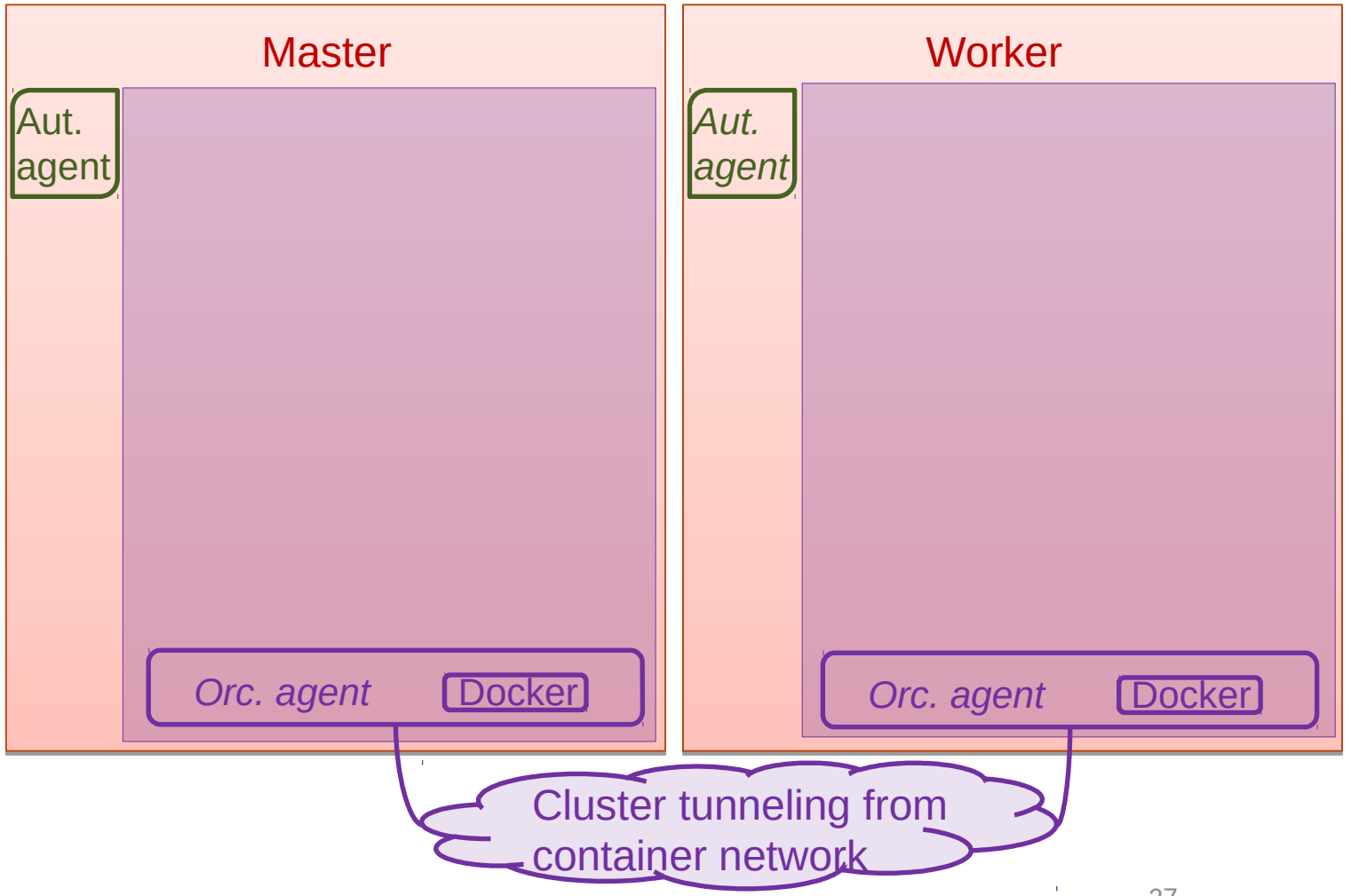
Under the hood (Micro overview)

What is on each server ?



Under the hood (Micro overview)

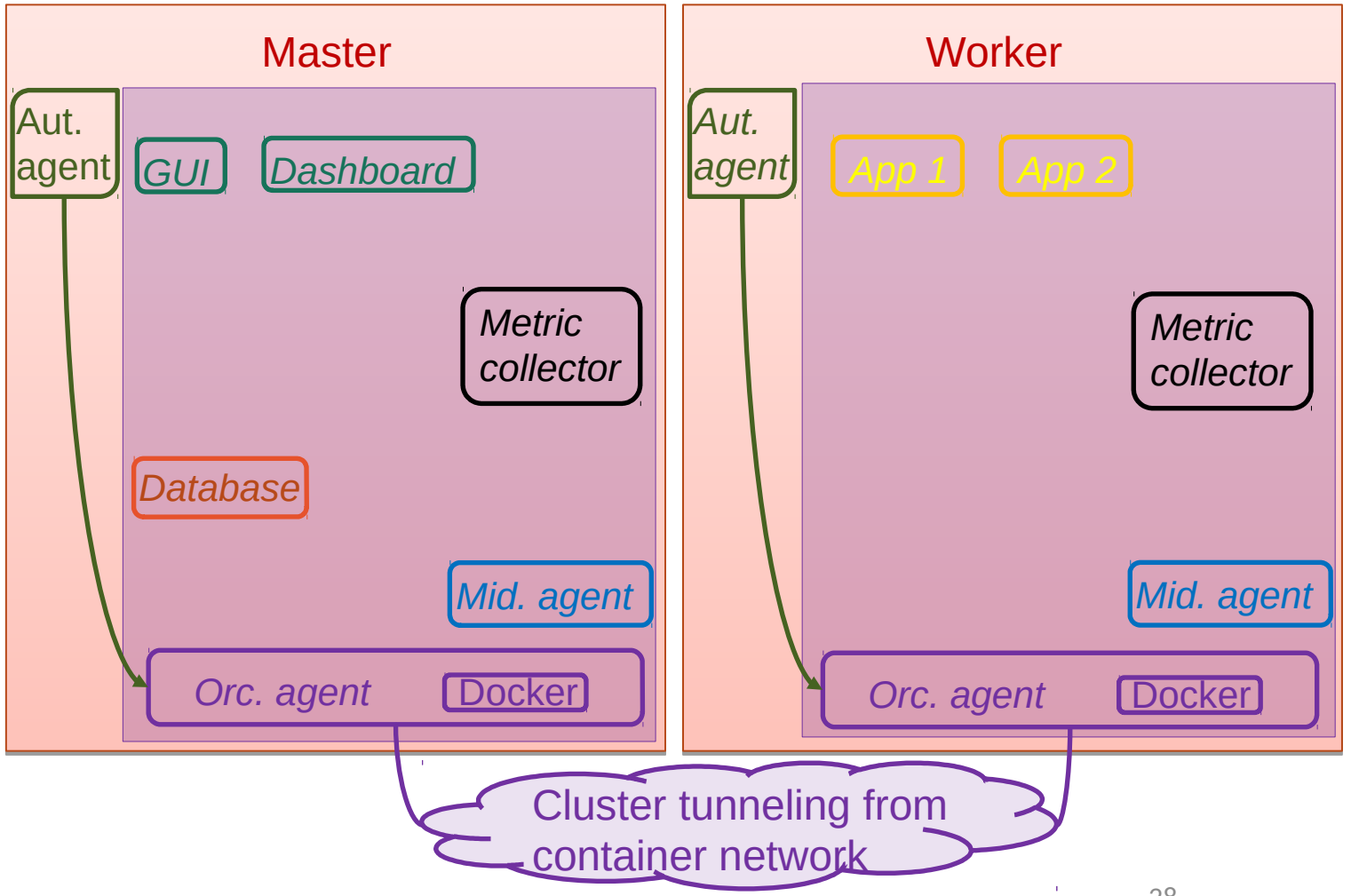
Docker enables virtual and closed environments for testing.



Under the hood (Micro overview)

Step 3

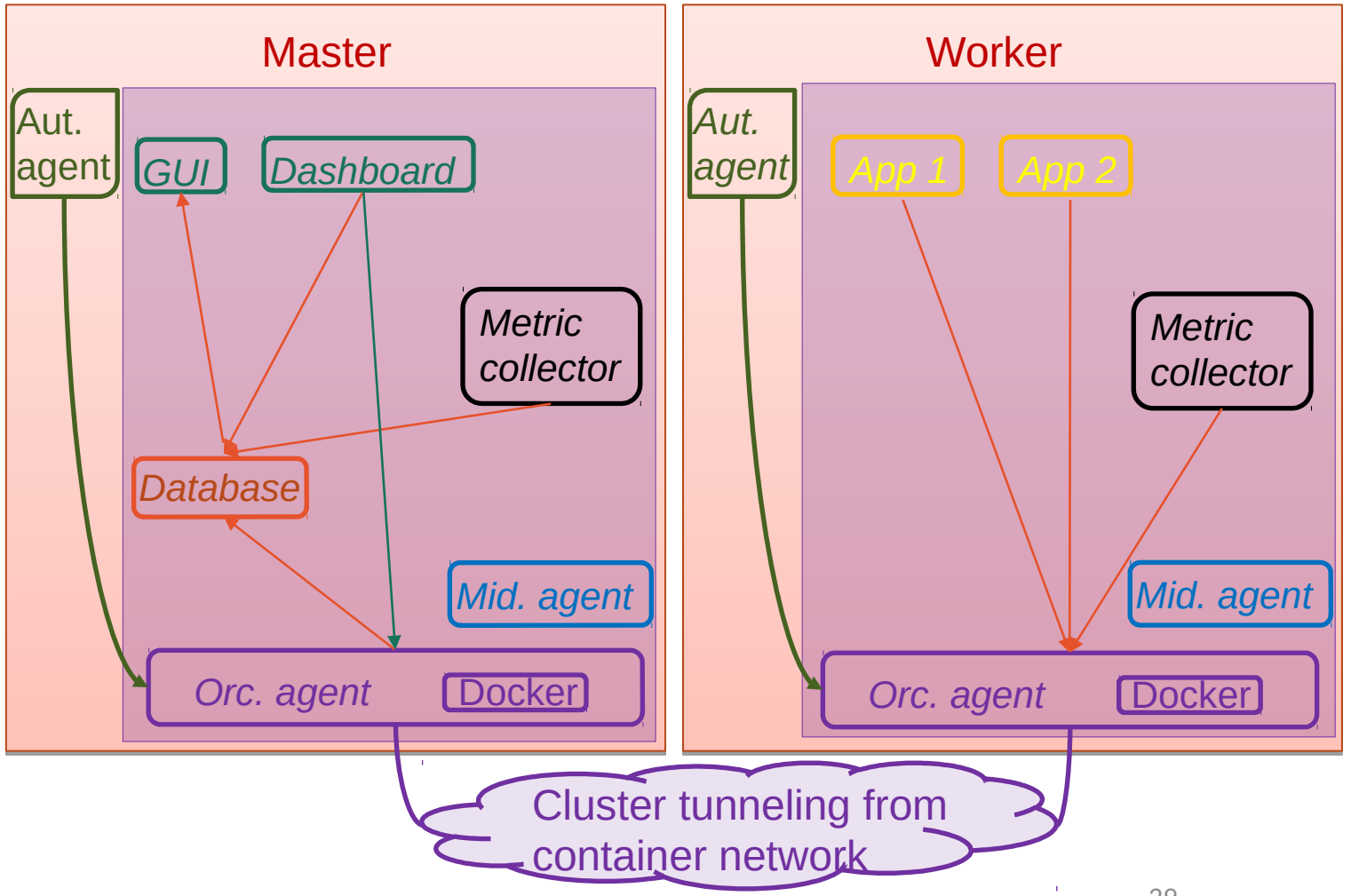
Install the required *modules* through automation



Under the hood (Micro overview)

Step 3

Install the required *modules* through automation



Provision

Task

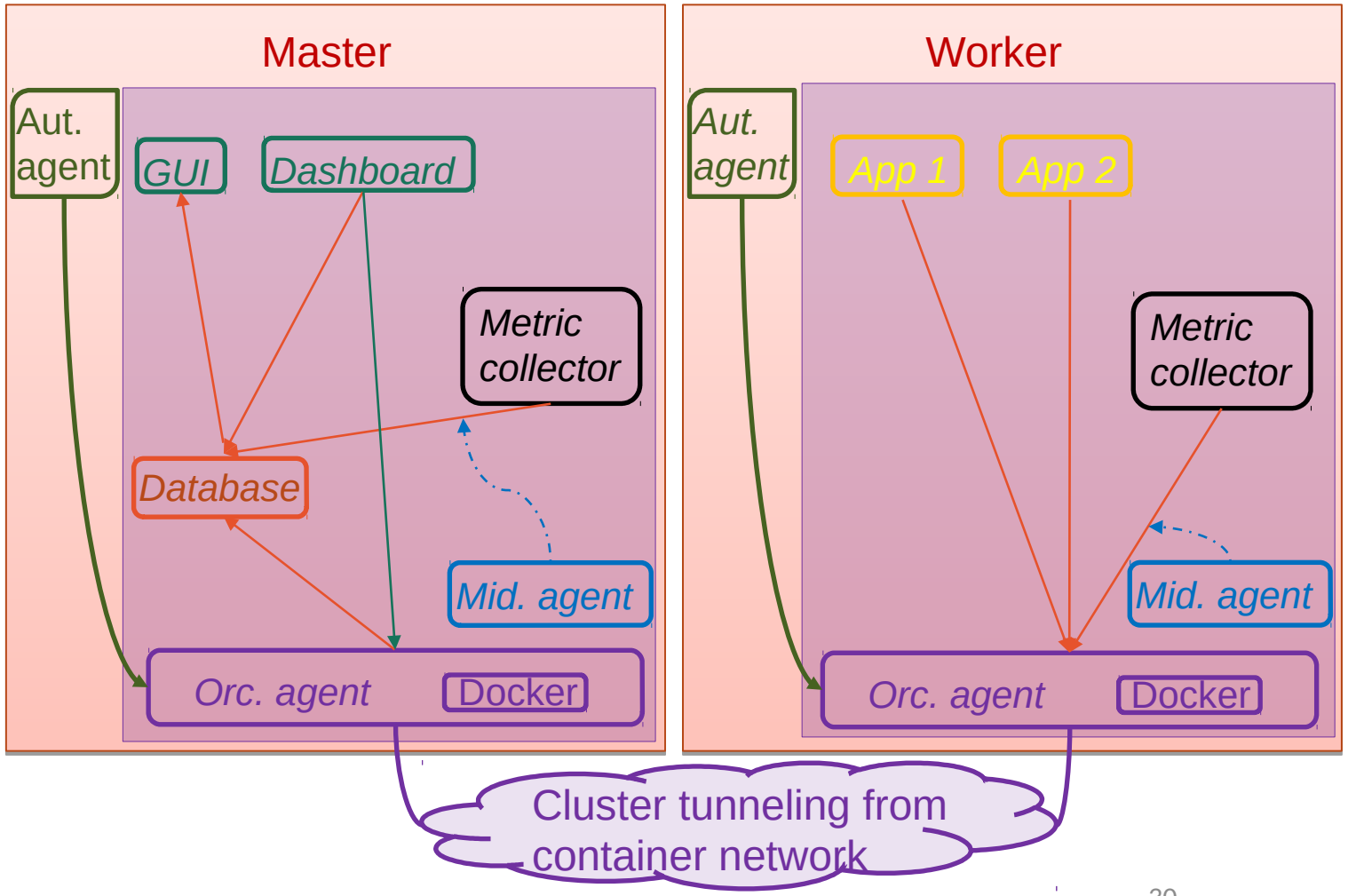
Under the hood (Micro overview)

Step 3

Install the required *modules* through automation

Step 4

Link additionnal *modules* with middleware



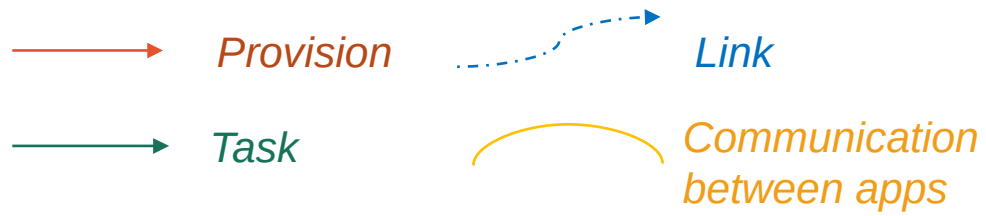
Under the hood (Micro overview)

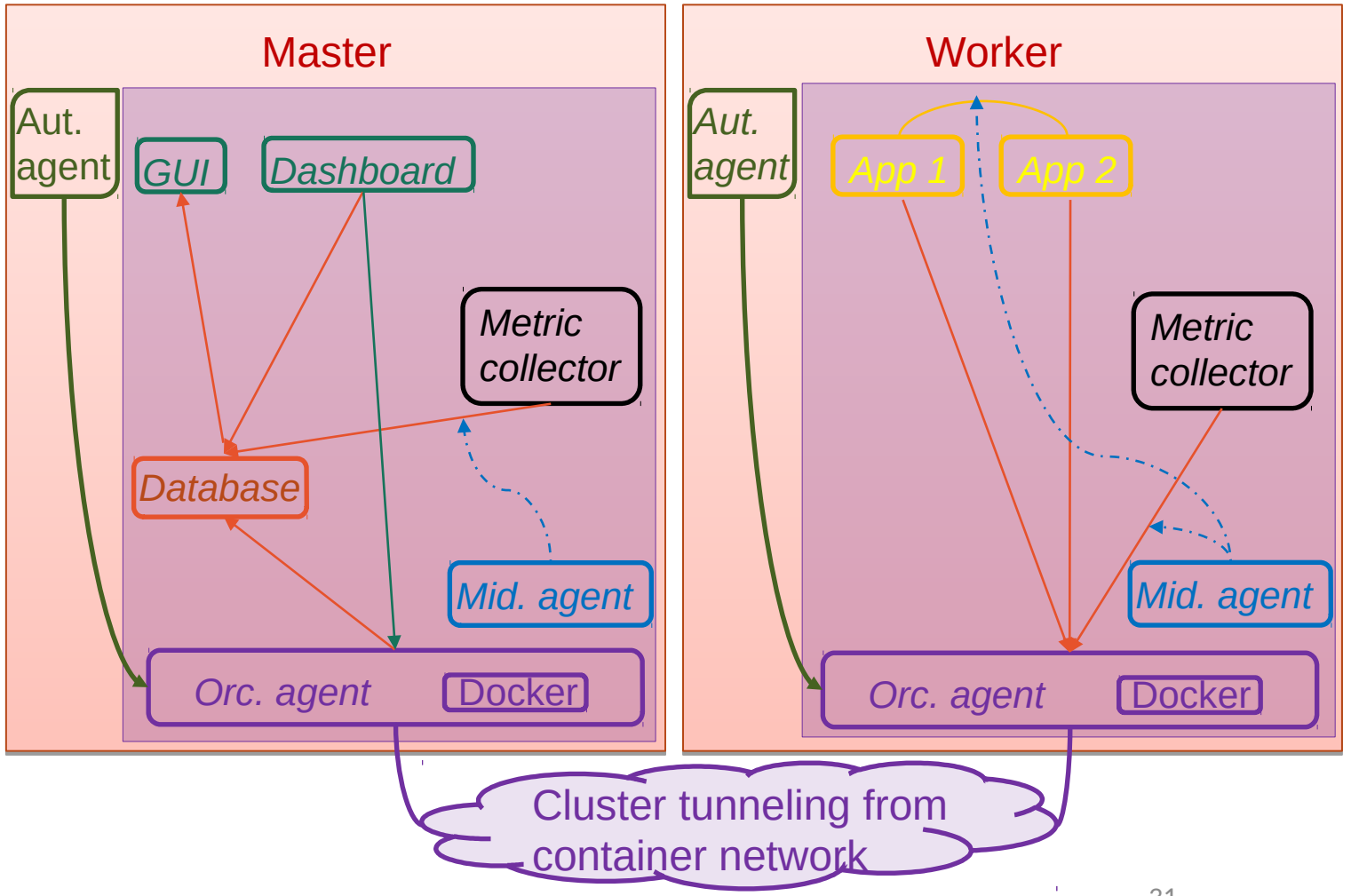
Step 3

Install the required *modules* through automation

Step 4

Link additionnal *modules* with middleware





Under the hood (Security)

- Your credentials

  Ansible Vault plays around the YAML variables to run its playbook.
  It encrypts sensitive variables and files behind a *vaulttext* by concatenating the ciphertext and a SHA256 digest.

- Your nodes

  The NMaaS runs in a cloud fashion, and thus trusts your own infrastructure. A VPN is recommended to ease the deployment.
  Kubernetes adds a layer for all API traffic and authentification, using x509 generated certificates and RBAC.
  Then, services select their own HTTPS endpoints to be exposed.

Agenda

Context

Overview

Under the hood

How to deploy

Conclusion

How to deploy

Step 1 : Set up the environment

Master

Worker

- Update your packages
- Get the OpenSSH client
- Get Python and its Pip version

Installer

- Exchange all the machines' SSH keys with the OpenSSH server
- Install the requirements.txt with Pip

How to deploy

Step 2 : List your nodes

Installer

- Populate the *inventory/hosts.yml* with their IP adresses

```
node1:
  ansible_host: 10.8.0.1
  ip: 10.8.0.1
  access_ip: 10.8.0.1
node2:
  ansible_host: 10.8.0.2
  ip: 10.8.0.2
  access_ip: 10.8.0.2
node3:
  ansible_host: 10.8.0.3
  ip: 10.8.0.3
  access_ip: 10.8.0.3
```

How to deploy

Step 2 : List your nodes

Installer

- Populate the *inventory/hosts.yml* with their IP adresses
- For each node, create its *vars* file in its *own* *inventory/hosts_vars/node1* folder

```
ansible_user: "{{ vault_ansible_user_node1 }}"
ansible_port: "{{ vault_ansible_port_node1 }}"
ansible_become_password: "{{ vault_ansible_become_password_node1 }}"
```

(Write exactly as is and switch the node's number)

How to deploy

Step 2 : List your nodes

Installer

- Populate the *inventory/hosts.yml* with their IP adresses
- For each node, create its *vars* file in its *own* *inventory/hosts_vars/node1* folder
- Encrypt your credentials in a *vault* file in the same folder with Ansible Vault

```
vault_ansible_user_node1: ssh_user
vault_ansible_become_password_node1: sudo_password
vault_ansible_port_node1: port_number
```

(Insert your real credentials here)

How to deploy

Step 3 : Deploy the platform

Installer

- Test the connection and credential authentification with Ansible

```
ansible all -i inventory/hosts.yml -m ping --ask-vault-pass
```

How to deploy

Step 3 : Deploy the platform

Installer

- Test the connection and credential authentification with Ansible
- Launch the deployment

```
ansible-playbook -i inventory/hosts.yml --become --become-user=root init.yml \
--ask-vault-pass  -e@inventory/host_vars/vault
```

How to deploy

Step 3 : Deploy the platform

Installer

- Test the connection and credential authentification with Ansible
- Launch the deployment
- Check if everything runs smoothly

```
sudo ./inventory/artifacts/kubectl.sh --kubeconfig inventory/artifacts/admin.conf get all --all-namespaces
```

Agenda

Context

Overview

Under the hood

How to deploy

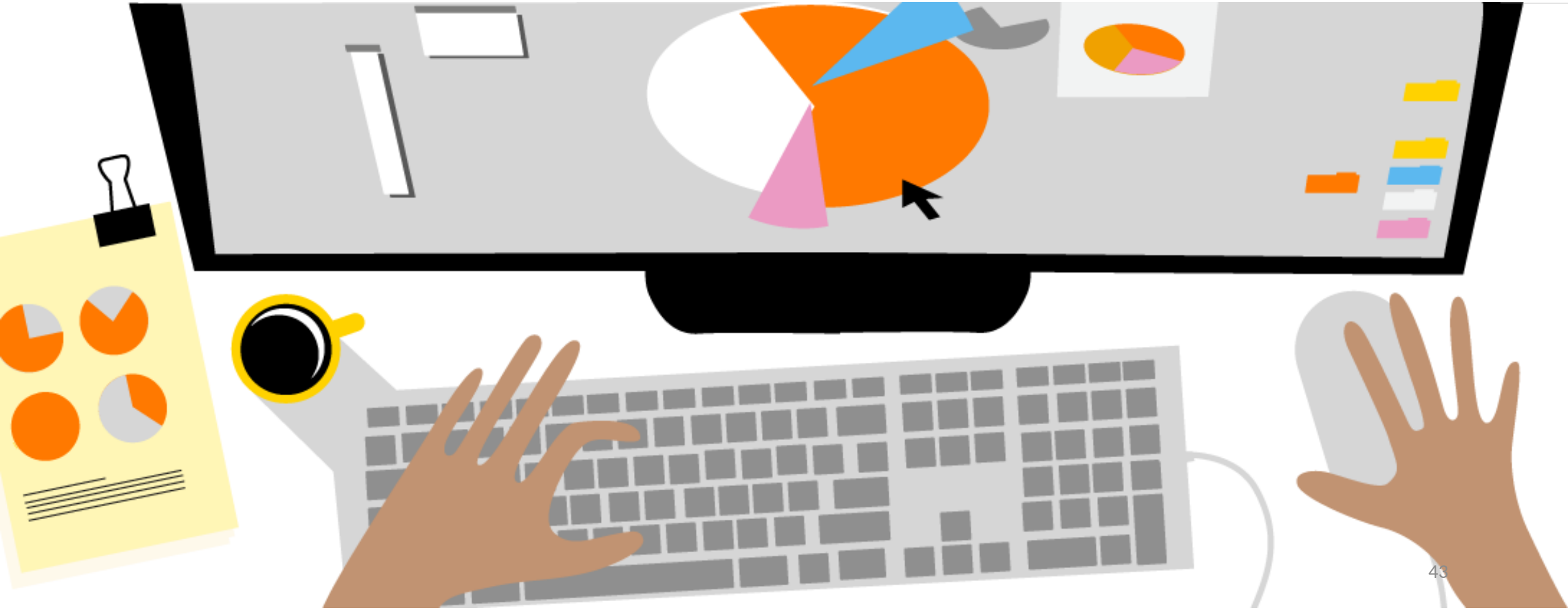
Conclusion

Conclusion

The NMaaS allows rationalizing network and service monitoring, while scaling and ensuring accurate measurements. Gains are numerous:

- Open source (https://github.com/Orange-OpenSource/NMaaS) and publicly available for the community to use and extend;
- Automation in deployment and use, no need to go on site to deploy new measurement applications → lockdown friendly;
- Easy and rapid integration of applications to the platform thanks to the catalog system and its container-based architecture (first catalog to be released soon);
- ACS-based scheduler that enables proper ressource allocation (to be released soon).

Thank you



References

- NMaaS : https://github.com/Orange-OpenSource/NMaaS
- Docker : https://www.docker.com
- Kubernetes : https://kubernetes.io
- Ansible : https://www.ansible.com
- Prometheus : https://prometheus.io
- Grafana : https://grafana.com
- Rancher : https://rancher.com
- Istio : https://istio.io
- Kiali : https://www.kiali.io
- Kubespray : https://kubespray.io
- Kube-Prometheus : https://github.com/coreos/kube-prometheus